UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
10/684,912	10/14/2003	Michael A. Stokke	MS301462.1 / MSFTP462US	3556

27195 7590 07/31/2009

TUROCY & WATSON, LLP
127 Public Square
57th Floor, Key Tower
CLEVELAND, OH 44114

EXAMINER	
AUGUSTINE, NICHOLAS	
ART UNIT	PAPER NUMBER
2179	

NOTIFICATION DATE	DELIVERY MODE
07/31/2009	ELECTRONIC

Please find below and/or attached an Office communication concerning this application or proceeding.

The time period for reply, if any, is set in the attached communication.

Notice of the Office communication was sent electronically on above-indicated "Notification Date" to the following e-mail address(es):

docket1@thepatentattorneys.com
hholmes@thepatentattorneys.com
lpasterchek@thepatentattorneys.com

Office Action Summary	Application No.	Applicant(s)	
	10/684,912	STOKKE ET AL.	
	Examiner	Art Unit	
	NICHOLAS AUGUSTINE	2179	

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE 3 MONTH(S) OR THIRTY (30) DAYS, WHICHEVER IS LONGER, FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

1)☒ Responsive to communication(s) filed on *29 April 2009*.

2a)☒ This action is **FINAL**. 2b)☐ This action is non-final.

3)☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

4)☒ Claim(s) *1-4,7-29 and 33* is/are pending in the application.

4a) Of the above claim(s) _____ is/are withdrawn from consideration.

5)☐ Claim(s) _____ is/are allowed.

6)☒ Claim(s) *1-4, 7-29, 33* is/are rejected.

7)☐ Claim(s) _____ is/are objected to.

8)☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

9)☐ The specification is objected to by the Examiner.

10)☐ The drawing(s) filed on _____ is/are: a)☐ accepted or b)☐ objected to by the Examiner.

Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).

11)☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. § 119

12)☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

a)☐ All b)☐ Some * c)☐ None of:

1.☐ Certified copies of the priority documents have been received.

2.☐ Certified copies of the priority documents have been received in Application No. _____.

3.☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

Attachment(s)

1) ☐ Notice of References Cited (PTO-892)
2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)
3) ☐ Information Disclosure Statement(s) (PTO/SB/08) Paper No(s)/Mail Date _____.
4) ☐ Interview Summary (PTO-413) Paper No(s)/Mail Date. _____.
5) ☐ Notice of Informal Patent Application
6) ☐ Other: _____.

DETAILED ACTION

A. This action is in response to the following communications: Amendment filed: 04/29/2009. This action is made **Final**.

B. Claims 1-4, 7-29 and 33 remain pending.

C. Rejection under 35 USC 101 for claims 1-4,7, 21 and 28 is withdrawn due to amendment.

Claim Rejections - 35 USC § 101

2. 35 U.S.C. 101 reads as follows:

> Whoever invents or discovers any new and useful process, machine, manufacture, or composition of matter, or any new and useful improvement thereof, may obtain a patent therefor, subject to the conditions and requirements of this title.

3. Claims 29 is rejected under 35 U.S.C. 101 because the claimed invention is directed to non-statutory subject matter. Claims are directed to a program per se as they are directed to a user interface, which as described in the specification is mere software; a computer program per se is not included in one of the statutory categories of invention and is believed to be non-statutory, more information about this matter is covered in the Annex IV of the Interim Guidelines for Subject matter Eligibility.

> http://www.uspto.gov/web/offices/pac/dapp/opla/preognotice/guidelines101_20051026.pdf

Claim Rejections - 35 USC § 103

4. The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

> (a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made.

5. This application currently names joint inventors. In considering patentability of the claims under 35 U.S.C. 103(a), the examiner presumes that the subject matter of the various claims was commonly owned at the time any inventions covered therein were made absent any evidence to the contrary. Applicant is advised of the obligation under 37 CFR 1.56 to point out the inventor and invention dates of each claim that was not commonly owned at the time a later invention was made in order for the examiner to consider the applicability of 35 U.S.C. 103(c) and potential 35 U.S.C. 102(e), (f) or (g) prior art under 35 U.S.C. 103(a).

6. Claims 1, 4,7-11, 14-22, 25, 27-29 and 33 are rejected under 35 U.S.C. 103(a) as being unpatentable over Schaefer (US Pub 2003/0084429), herein referred to as Schaefer, in view of Dewhurst et al. (US 6,430,609) herein referred to as Dewhurst.

As for claim 1, Schaefer discloses user interface automation system (fig.2, label 210, 220 and 230) comprising: a processor that executes the following computer executable components stored in memory (par.40); an input component that receives a request (par [0037], lines 6-8); a navigation component that receives the request from the input component (par [0037], lines 3-6) and facilitates simulating a user interface associated with an automation component (par [0038], lines 1-3 and 5-8; par [0041]; par [0051]; par

[0054]-[0055]; par [0071]), a map information store comprising program flow information wherein one or more section names divide the map information store into specific page data such that each of the section name references a specific page of the automation component (par [0041], lines 1-3; par [0043]; par [0045]; par.[94-96]); a command information store disparate from the map information store that comprises the section names associated with the specific page of the automation component and one or more actions to execute for the specific page (figure 2, wherein depicted are individual components that are separate from one another; par.; 42,49,97-99]); wherein the navigation component facilitates simulating the user interface based at least upon information stored in the command information store and the map information store, wherein the map information store (par.[53-55], the command information store and executables are stored separately (par [0041], lines 8-14; par [0043]; par [0045]).

Schaefer does not expressly teach *the navigation component further employs a global information store and facilitates a global variable replacement from a single location and sharing of a common program flow among a plurality of users* in great detail. Schaefer teaches that their system takes the navigation component along with other components and relevant information is stored on secondary storage. Secondary storage as known in the art is commonly placed outside of a local computing area (remote server, "physically not connected to the computer running content on primary storage). It is also commonly well known that secondary storage can be locally as well. Schaefer does not distinguish between the two possibilities (paragraph 43 and 45). However Schaefer makes mention of a network interface module 150 which is used in the system and is

defined as "may include hardware ("secondary storage" could be expressed by this) and software for sending and receiving data over a network, and may be user, for example, in testing a software program that has a client/ server architecture". Of course it would have been obvious to one of ordinary skill in the art at the time of the invention was made to recognize that secondary storage is used for the storage of the navigation component which is globally shared among a plurality of users in a client/server architecture as suggested by Schaefer (paragraph 43,45-46).

Schaefer does not expressly teach *the navigation component further employs a global information store and facilitates a global variable replacement from a single location and sharing of a common program flow among a plurality of users; the navigation component further modifies the user interface automation without recompiling executables by modifying one or more of the map information store or the command information store* in complete details without one of ordinary skill in the art to obviously make the connection. However in the same field of endeavor Dewhurst teaches *the navigation component further employs information stored in a global information store when a global variable is encountered in the command information store and facilitates a global variable replacement from a single location and sharing of a common program flow among a plurality of users the navigation component further modifies the user interface automation without recompiling executables by modifying one or more of the map information store or the command information store* (column 4, lines 33-65 and col.11, lines 6-47; figure 5; wherein depicted are plurality of users accessing information from a remote location "server", wherein information is a global information store where

connected users can replace variables in the global information store. The replacement information is updated for other connected users from other single locations (clients). Of course it would have been obvious to one of ordinary skill in the art at the time of the invention to include Dewhurst's detailed view of global sharing into Schaefer broad view of global sharing, this is true because Dewhurst and Schaefer both teach systems and methods of automation of software presented to the user using a repository connected to a client.

In summary Dewhurst teaches plurality of users accessing information from a remote location "server", wherein information is a global information store where connected users of the system can replace variables (data) in the global information store. The replacement information is updated so for other connected users from other single locations (clients "PC") can view the updated data, thus execution is not halted or restarted that requires a recompile of the software system. This makes it possible for users to edit global information (information stored globally for a plurality of users can view/edit information) and then when a user request part or all of the global information the most current version of that information is retrieved from the server to be presented to the user, thus providing a dynamic changing information source that does not need to be recompiled to offer different results of the end user presentation (col.4, lines 33-65; col.11, lines 6-47). Further Schaefer describes how the test engine "navigates" through the testing of a software program which is automated providing to the user an absence of having to create test scripts (par.37). The test engine drives through the software program testing it along the way, thus acts like a navigation component that facilitates

simulated user interface associated with an automation component. The GUI translator component translates one or more GUI maps into a set of database tables, these tables are read by the test engine for navigation through the software program, thus the test engine is a navigation component which reads automation test scripts produced by the system which allows users whom are inexperience with writing testing scripts that navigate through a software program to have automated testing scripts produced and ran by the system (par.37,38,40, 41-42, 48-55 and 71-72).

As claim 3, Schaefer further discloses the map information store comprises a text-based file (par [0048]), lines 11-14).

As claim 4, Schaefer does not specifically disclose the configuration information store comprises a text-based file. However, Schaefer discloses a text-based file (par [0047], that HTML is a text based file; par [0048]), lines 11-14). Therefore, it would have been obvious to one of ordinary skill in the art at the time the invention to use a text-based file for the command information store in order to store and organize information about a window and objects on the window, such as text fields, boxes, buttons, menus, etc., and making it easy to edit using available software programs installed with most operating systems (e.g., text editing program).

As claim 7, Schaefer further discloses at least one of the map information store and the configuration information store comprise at least one alias name (par [0050], lines 5-6).

As claim 8, Schaefer further discloses the navigation component further stores error information in a log information store (par [0115]-[0116]).

As claim 9, Schaefer further inherently discloses the navigation component further stores information associated with the request in a log information store (par [0115]-[0116]- It should be recognized that the steps of monitoring the results of the execution of the program, test engine component 170 may generate a text-based log file, and store information about the results of the execution including information about windows, .GUI map for each window, objects on the window, Actions that were taken, a status of whether the test case passed or failed, TimeStart and TimeStop for a window action, etc.).

As claim 10, Schaefer further discloses the navigation component iterates through information stored in the command information store (par [0041], lines 8-14; par [0043]; par [0045]), performs the indicated operation (fig. 12; par [0115]) and stores information associated with the indicated operation in the log information store (par [0115]-[0116]).

As claim 11, Schaefer further discloses the navigation component stores error information in the log information store (par [0115]-[0116]).

As claim 14, Schaefer further discloses the input component receives a command line invocation (par [0012]), lines 1-5).

As claim 15, Schaefer further discloses the map information store comprises (fig. 8c, label 825c; par [0081], lines 6-9) a page identifier (fig. 8c, label 820c; par [0081], lines 6-9).

As claim 16, Schaefer discloses the page identifier comprising a label for a control (fig. 11, label 1100; par [0100], lines 5-10), the page identifier further uniquely identifying a page (fig. 8c, label 820c; par [0081], lines 6-9; fig. 8b, label 890b and 810a); par [0079], lines 1-13).

As claim 17, Schaefer further discloses the page identifier comprising a control type (fig. 11, label 1100; par [0100], lines 5-10).

As claim 18, Schaefer further discloses the control type is at least one of button, combo, list, scroll, static, radio and check type (fig. 11, label 1100 and 1140; par [0101], lines 1-5).

As claim 19, Schaefer further inherently discloses information stored in the command information store can be modified by at least one of a front-end user interface application, scripting, a batch file or a text editor (par [0093] that Schaefer discloses the associated GUI map can be edit using a GUI map editor, since the command information store, which is associated with the GUI map, therefore it can be modified with the same concept).

As claim 20, Schaefer further discloses the command information store comprising a section name, the section name corresponding to information stored in the map information store (fig. 13).

As claim 21, Schaefer further discloses the command information store storing information associated with at least one of a function key or a control key simulation (fig. 13).

As claim 22, Schaefer discloses a method of automating user interface (fig.2, label 210, 220 and 230) comprising: employing a processor to execute computer executable instructions stored in memory to perform the following acts (par.[40]): receiving a request invoking a user interface automation system; storing information related to specific pages of the automation system and corresponding acts to execute for the pages in a disparate map information store and a command information store respectively (par.53,94-96) receiving mapping information from the map information store comprising one or more section names that reference specific pages of the automation system (par [0041], lines 5-8,par.42); receiving command information from a command information store comprising specific section names corresponding to information stored in the map information store and information associated with commands to be executed for respective pages of the automation system (par [0041], lines 8-14; par.42-43,96-99); simulating a user interface (par [0038], lines 1-3 and 5-8; par [0041]; par [0051]; par [0054]-[0055]; par [0071]), based at least in part, upon information stored in the map information store (par [0041], lines 1-3; par [0043]; par [0045]) and the command information store employing information stored in the global information store when a global variable is encountered in the command information store (par [0041], lines 8-14; par [0043]; par [0045]).

Schaefer does not specifically mention retrieving global information from a global information store; modifying the user interface automation utilizing existing compiled executables upon modification of one or more of the map information store or the command information store by storing data, commands and executables separately.

However in the same field of endeavor Dewhurst teaches *retrieving global information from a global information store; modifying the user interface automation* utilizing existing compiled executables upon modification of one or more of the map information store or the command information store *without recompilation of executables by storing data, commands and executables separately* (column 4, lines 33-65 and figure 5; wherein depicted are plurality of users accessing information from a remote location "server", wherein information is a global information store where connected users can replace variables in the global information store. The replacement information is updated for other connected users from other single locations (clients). Of course it would have been obvious to one of ordinary skill in the art at the time of the invention to include Dewhurst's detailed view of global sharing into Schaefer broad view of global sharing, this is true because Dewhurst and Schaefer both teach systems and methods of automation of software presented to the user using a repository connected to a client.

In summary Dewhurst teaches plurality of users accessing information from a remote location "server", wherein information is a global information store where connected users of the system can replace variables (data) in the global information store. The replacement information is updated so for other connected users from other single locations (clients "PC") can view the updated data, thus execution is not halted or restarted that requires a recompile of the software system. This makes it possible for users to edit global information (information stored globally for a plurality of users can view/edit information) and then when a user request part or all of the global information the most current version of that information is retrieved from the server to be presented

to the user, thus providing a dynamic changing information source that does not need to be recompiled to offer different results of the end user presentation (col.4, lines 33-65; col.11, lines 6-47). Further Schaefer describes how the test engine "navigates" through the testing of a software program which is automated providing to the user an absence of having to create test scripts (par.37). The test engine drives through the software program testing it along the way, thus acts like a navigation component that facilitates simulated user interface associated with an automation component. The GUI translator component translates one or more GUI maps into a set of database tables, these tables are read by the test engine for navigation through the software program, thus the test engine is a navigation component which reads automation test scripts produced by the system which allows users whom are inexperience with writing testing scripts that navigate through a software program to have automated testing scripts produced and ran by the system (par.37,38,40, 41-42, 48-55 and 71-72).

As claim 24, Schaefer further discloses a computer readable medium (par [0045], lines 1-2) having stored thereon computer executable instructions for carrying out the, method of claim 22 (par [0040], lines 1-5).

As claim 25, Schaefer discloses a method of automating user interface (fig.2, label 210, 220 and 230) comprising: employing a processor to execute computer executable instructions stored in memory to perform the following acts (par.40): receiving a request for executing a user interface automation component (par.41-43;53) retrieving mapping information from a map file comprising at least a section name and a page identifier for pages generated by the automation component (par [0041], lines 5-8); retrieving commands to be executed for the pagesfrom a command file (par [0041], lines 8-14, par.42-43,53-54);obtaining a section name from the command file (fig. 8c, label 825c; par [0081], lines 6-9); retrieving page identification information from the map file associated with the section name (fig. 8c, label 820c; par [0081], lines 6-9; fig. 8b, label 890b and 810a); par [0079], lines 1-13); retrieving section data for section associated with the section name from the command file (fig.10); and, executing the commands associated with the retrieved section data; separately storing at least one of: the map file, the command file; or the compiled executables (fig. 1 and 11). Schaefer does not specifically mention employing information stored in the global file when a global variable is encountered in the command file; retrieving global information from a global information store; modifying the user interface automation without recompilation of executables by storing data, commands and executables separately. However in the same field of endeavor Dewhurst teaches *retrieving global information from a global information store; modifying the user interface automation without recompilation of executables when one or more of the command file and/or map file are modified while maintaining compilation of executables* (column 4, lines 33-65; col.11, lines 6-47 and

figure 5; wherein depicted are plurality of users accessing information from a remote location "server", wherein information is a global information store where connected users can replace variables in the global information store (command file). The replacement information is updated for other connected users from other single locations (clients). Of course it would have been obvious to one of ordinary skill in the art at the time of the invention to include Dewhurst's detailed view of global sharing into Schaefer broad view of global sharing, this is true because Dewhurst and Schaefer both teach systems and methods of automation of software presented to the user using a repository connected to a client.

As claim 27, Schaefer further discloses a computer readable medium (par [0045], lines 1-2) having stored thereon computer executable instructions for carrying out the method of claim 25 (par [0040], lines 1-5).

As claim 28; Schaefer discloses an automation system for a user interface (fig.2, label 210,
220 and 230) comprising: a processor that executes the following computer executable components stored in memory; an input component that receives a request associated with generating a user interface (par [0037], lines 6-8);, a map information store comprising program flow information wherein one or more section names divide the

map information store into specific page data such that each of the section name references a specific page of the automation component (par [0041], lines 1-3; par [0043]; par [0045]; par.[94-96]); a command information store disparate from the map information store that comprises the section names associated with the specific page of the automation component and one or more actions to execute for the specific page (figure 2, wherein depicted are individual components that are separate from one another; par.; 42,49,97-99]);a navigation component that receives the request from the input component (par [0037], lines 3-6) and facilitates generating the user interface associated with an automation component (par [0038], lines 1-3 and 5-8; par [0041]; par [0051]; par [0054]-[0055]; par [0071]), based at least in part, upon information stored in a map information store (par [0041], lines 1-3; par [0043]; par [0045]) and information stored in a command information store, the map information store, the command information store and executables are stored separately (par [0041], lines 8-14; par [0043]; par [0045], figure 1). Schaefer does not expressly teach *the navigation component further employs a global information store and facilitates a global variable replacement from a single location and sharing of a common program flow among a plurality of users* in great detail. Schaefer teaches that their system takes the navigation component along with other components and relevant information is stored on secondary storage. Secondary storage as known in the art is commonly placed outside of a local computing area (remote server, "physically not connected to the computer running content on primary storage). It is also commonly well known that secondary storage can be locally as well. Schaefer does not distinguish between the two

possibilities (paragraph 43 and 45). However Schaefer makes mention of a network interface module 150 which is used in the system and is defined as "may include hardware ("secondary storage" could be expressed by this) and software for sending and receiving data over a network, and may be user, for example, in testing a software program that has a client/ server architecture". Of course it would have been obvious to one of ordinary skill in the art at the time of the invention was made to recognize that secondary storage is used for the storage of the navigation component which is globally shared among a plurality of users in a client/server architecture as suggested by Schaefer (paragraph 43,45-46).

Schaefer does not expressly teach *the navigation component further employs a global information store and facilitates a global variable replacement from a single location and sharing of a common program flow among a plurality of users; the navigation component further modifies the user interface automation without recompiling executables by modifying one or more of the map information store or the command information store* in complete details without one of ordinary skill in the art to obviously make the connection. However in the same field of endeavor Dewhurst teaches *the navigation component further employs information stored in a global information store when a global variable is encountered in the command information store and facilitates a global variable replacement from a single location and sharing of a common program flow among a plurality of users the navigation component further modifies the user interface automation without recompiling executables by modifying one or more of the map information store or the command information store* (column 4, lines 33-65 and

col.11, lines 6-47; figure 5; wherein depicted are plurality of users accessing information from a remote location "server", wherein information is a global information store where connected users can replace variables in the global information store. The replacement information is updated for other connected users from other single locations (clients). Of course it would have been obvious to one of ordinary skill in the art at the time of the invention to include Dewhurst's detailed view of global sharing into Schaefer broad view of global sharing, this is true because Dewhurst and Schaefer both teach systems and methods of automation of software presented to the user using a repository connected to a client.

As claim 29, Schaefer discloses an automation system for a user interface (fig.2, label 210,220 and 230) comprising: means for receiving a request in connection with simulating a user interface; means for storing mapping information comprising one or more section names that reference specific pages of the user interfaces (par [0037], lines 6-8); means for storing command information disparate from the mapping information storing means, the command information storing means comprising specific section names corresponding to the one or more section names stored in the mapping means, the command information comprises at least commands to be executed for respective pages of the user interface (par.42-43,53-56,96-99);means for simulating user interface associated with an automation component (par [0038], lines 1-3 and 5-8; par [0041]; par [0051]; par [0054]-[0055]; par [0071]), based at least in part, upon information stored in the mapping information storing means and the command information storing means (par [0041], lines 1-3; par [0043]; par [0045],par [0041], lines

8-14) the means for simulating receiving the request from the means for receiving (par [0037], lines 3-6).

Schaefer does not expressly teach *the navigation component further employs a global information store and facilitates a global variable replacement from a single location and sharing of a common program flow among a plurality of users* in great detail. Schaefer teaches that their system takes the navigation component along with other components and relevant information is stored on secondary storage. Secondary storage as known in the art is commonly placed outside of a local computing area (remote server, "physically not connected to the computer running content on primary storage). It is also commonly well known that secondary storage can be locally as well. Schaefer does not distinguish between the two possibilities (paragraph 43 and 45). However Schaefer makes mention of a network interface module 150 which is used in the system and is defined as "may include hardware ("secondary storage" could be expressed by this) and software for sending and receiving data over a network, and may be user, for example, in testing a software program that has a client/ server architecture". Of course it would have been obvious to one of ordinary skill in the art at the time of the invention was made to recognize that secondary storage is used for the storage of the navigation component which is globally shared among a plurality of users in a client/server architecture as suggested by Schaefer (paragraph 43,45-46).

For further evidence for one of ordinary skill in the art, Schaefer does not expressly teach *a means for sharing a common program flow among a plurality of users based, at least in part, upon replacing a global variable in the command information store with*

corresponding data from a global information store in complete details without one of ordinary skill in the art to obviously make the connection. However in the same field of endeavor Dewhurst teaches *means for sharing a common program flow among a plurality of users based, at least in part, upon replacing a global variable in the command information store with corresponding data from a global information store and means for modifying the user interface automation when one or more of the mapping information storing means or the command information storing means is modified, while maintaining compiled executables as mapping information, command information and the compiled executables are stored separately* (column 4, lines 33-65 and figure 5; wherein depicted are plurality of users accessing information from a remote location "server", wherein information is a global information store where connected users can replace variables in the global information store. The replacement information is updated for other connected users from other single locations (clients). Of course it would have been obvious to one of ordinary skill in the art at the time of the invention to include Dewhurst's detailed view of global sharing into Schaefer broad view of global sharing, this is true because Dewhurst and Schaefer both teach systems and methods of automation of software presented to the user using a repository connected to a client.

In summary Dewhurst teaches plurality of users accessing information from a remote location "server", wherein information is a global information store where connected users of the system can replace variables (data) in the global information store. The replacement information is updated so for other connected users from other single locations (clients "PC") can view the updated data, thus execution is not halted or

restarted that requires a recompile of the software system. This makes it possible for users to edit global information (information stored globally for a plurality of users can view/edit information) and then when a user request part or all of the global information the most current version of that information is retrieved from the server to be presented to the user, thus providing a dynamic changing information source that does not need to be recompiled to offer different results of the end user presentation (col.4, lines 33-65; col.11, lines 6-47). Further Schaefer describes how the test engine "navigates" through the testing of a software program which is automated providing to the user an absence of having to create test scripts (par.37). The test engine drives through the software program testing it along the way, thus acts like a navigation component that facilitates simulated user interface associated with an automation component. The GUI translator component translates one or more GUI maps into a set of database tables, these tables are read by the test engine for navigation through the software program, thus the test engine is a navigation component which reads automation test scripts produced by the system which allows users whom are inexperience with writing testing scripts that navigate through a software program to have automated testing scripts produced and ran by the system (par.37,38,40, 41-42, 48-55 and 71-72).

As claim 33, Schaefer does not specifically disclose the configuration information store/ data and commands associated with program flow are stored in a text-based file.

However, Schaefer discloses a text-based file (par [0047], that HTML is a text based file; par [0048]), lines 11-14). Therefore, it would have been obvious to one of ordinary skill in the art at the time the invention to use a text-based file for the command information store in order to store and organize information about a window and objects on the window, such as text fields, boxes, buttons, menus, etc., and making it easy to edit using available software programs installed with most operating systems (e.g., text editing program).

7. Claim 2 is rejected under 35 U.S.C. 103(a) as being unpatentable over Schaefer in view of Dewhurst in further view of Minard (US Patent 6,247,020).

As claim 2, Schaefer does not teach the automation component is a wizard. However, Minard teaches the automation component is a wizard (fig. 4A; col. 3, lines 27-31; col. 6, the image showing the wizard menu along with the description of functions; col. 8, lines 41-51) Therefore, it would have been obvious to one of ordinary skill in the art at the time the invention was made to modify Schaefer by using a wizard as the automation component as taught by Minard in order to improve the user interface that simplifies the job by removing numerous windows and consolidating all the functions into one unified window for the user interface with to design, edit and debug allowing the user to activate by the push of a button (Minard: col. 3, lines 20-33).

8. Claims 12-13, 23 and 26 are rejected under 35 U.S.C. 103(a) as being unpatentable over Schaefer in view of Dewhurst in further view of Zimniewiez et al. (US Patent 6,744,450), hereinafter "Zimniewiez"

As claim 12, Schaefer does not teach the input component performs input validation upon the request and provides error information if the request is invalid. Schaefer does teach the basic principle and concept (par. [0110], that by when test data is to be entered into a text field, test engine component 170 may call an insert_text function. Software controller component 173 may transmit an appropriate instruction to the software program 185 to input the data into the object and may return the result of the processing of the instruction by the software program 185 to test engine component 170). However, Zimniewiez teaches the input component performs input validation upon the request (col. 9, lines 36-38) and provides error information if the request is invalid (col. 8, lines 13-17). Therefore, it would have been obvious to one of ordinary skill in the art at the time the invention was made to modify Schaefer by performing input validation in the input component upon the request and provides error information if the request is invalid as taught by Zimniewiez in order to order to provide the user with an indication the process is invalid and provides the user immediate feedback to initiate troubleshooting the cause of the invalid function/command (col. 8, lines 43-46).

As claim 13, Schaefer does not teach a graphical message is displayed to a user of the system, the graphical message being based, at least in part, upon the error information from the input component. However, Zimniewiez teaches a graphical message is displayed to a user of the system, the graphical message being based, at least in part, upon the error information from the input component (col. 7, lines 22-24). Therefore, it would have been obvious to one of ordinary skill in the art at the time the invention was made to modify Schaefer by displaying a graphical message to the user of the system, the graphical message being based, at least in part, upon the error information from the input component as taught by Zimniewiez in Order to provide the user with an indication the process is invalid and provides the user immediate feedback to initiate troubleshot the cause of the invalid function (Zimniewiez; col. 10, lines 11-15).

As claim 23, Schaefer does not teach storing information in a log information store, if an error is detected performing the simulated user interface. However, Zimniewiez teaches storing information in a log information store (col. 11, lines 52-56), if an error is detected performing the simulated user interface (fig. 4a, label 124 and 136; col. 7, lines 19-21; col. 8, lines 59-60). Therefore, it would have been obvious to one of ordinary skill in the art at the time the invention was made to modify Schaefer by storing information in a log information store, if an error is detected performing the simulated user interface as taught by Zimniewiez in order to provide the user with an indication the process is invalid and provides the user immediate feedback to initiate troubleshot the cause of the invalid function (Zimniewiez; col. 10, lines 11- 15).

As claim 26, Schaefer does not teach storing information in a log file, if an error is detected performing the action. However, Zimniewiez teaches storing information in a log file (col. 11, lines 52-56), if an error is detected performing the action (fig. 4a, label 124 and 136; col. 7, lines 19-21; col. 8, lines 59-60). Therefore, it would have been obvious to one of ordinary skill in the art at the time the invention was made to modify Schaefer by storing information in a log file, if an error is detected performing the action as taught by 7imniewiez in order to provide the user with an indication the process is invalid and provides the user immediate feedback to initiate troubleshot the cause of the invalid function (Zimniewiez; col. 10, lines 11-15).

(Note:) **It is noted that any citation to specific, pages, columns, lines, or figures in the prior art references and any interpretation of the references should not be considered to be limiting in any way. A reference is relevant for all it contains and may be relied upon for all that it would have reasonably suggested to one having ordinary skill in the art. In re Heck, 699 F.2d 1331, 1332-33, 216 USPQ 1038, 1039 (Fed. Cir. 1983) (quoting In re Lemelson, 397 F.2d 1006,1009, 158 USPQ 275, 277 (CCPA 1968)).**

Response to Arguments

Applicant's arguments filed 04/29/2009 have been fully considered but they are not persuasive.

A1. Applicant has specifically argued against the newly added limitation.

R1. After careful consideration the Examiner as not agreed with the attempt to overcome the prior art and provided new claim analysis regarding the newly added limitation into the claims rejected above in the 103 rejections.

Conclusion

THIS ACTION IS MADE FINAL. Applicant is reminded of the extension of time policy as set forth in 37 CFR 1.136(a).

A shortened statutory period for reply to this final action is set to expire THREE MONTHS from the mailing date of this action. In the event a first reply is filed within TWO MONTHS of the mailing date of this final action and the advisory action is not mailed until after the end of the THREE-MONTH shortened statutory period, then the shortened statutory period will expire on the date the advisory action is mailed, and any extension fee pursuant to 37 CFR 1.136(a) will be calculated from the mailing date of the advisory action. In no event, however, will the statutory period for reply expire later than SIX MONTHS from the mailing date of this final action.

Inquires

Any inquiry concerning this communication or earlier communications from the examiner should be directed to Nicholas Augustine whose telephone number is 571-270-1056 and fax is 571-270-2056. The examiner can normally be reached on Monday - Friday: 9:30am- 5:00pm Eastern.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Weilun Lo can be reached on 571-272-4847. The fax phone number for the organization where this application or proceeding is assigned is 571-273-8300.

Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see http://pair-direct.uspto.gov. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free). If you would like assistance from a USPTO Customer Service Representative or access to the automated information system, call 800-786-9199 (IN USA OR CANADA) or 571-272-1000.

/Steven B Theriault/
Primary Examiner, Art Unit 2179

/Nicholas Augustine/
Examiner
Art Unit 2179
July 21, 2009